MAY 11–12

ARSENAL

eBPFShield

Advanced IP-Intelligence and DNS Monitoring using eBPF

SAGAR BHURE

$whoami

SAGAR BHURE

- Software Security Engineer at F5
- Project Lead at OWASP
- OWASP CFP/CFT Reviewer

Talks

- ML for API Security at APISecure US
- Security for ML at Null India

linkedin.com/in/sagarbhure

github.com/sagarbhure

sagarbhure.com

Agenda

- ❑ Introduction
- ❑ eBPF Overview
- ❑ Enter eBPFShield

$whois eBPFShield

- eBPFShield is an open-source kernel-level security tool for network observability.
- Real-time security monitoring for businesses and individuals.

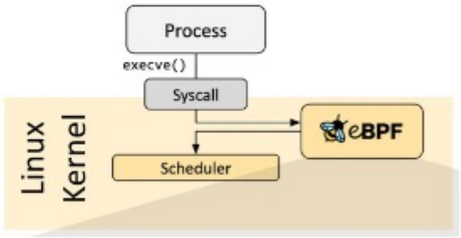
@sagarbhure.github.io/eBPFShield

$why eBPFShield

- Core OS changes are challenging.
- Kernel code changes need expertise.
- Kernel modules have high maintenance and compatibility issues.
- eBPF avoids kernel modification for extensions.
- eBPFShield = real-time network protection.
- eBPFShield prevents various exploit methods.
- eBPFShield improves security without kernel changes.

OVERVIEW - eBPF


```
function hello() {
  alert('hello')
}
</script>
<form onsubmit="return false;">
  <input type="submit"
         name="hello"
         onclick="hello()">
</form>
```

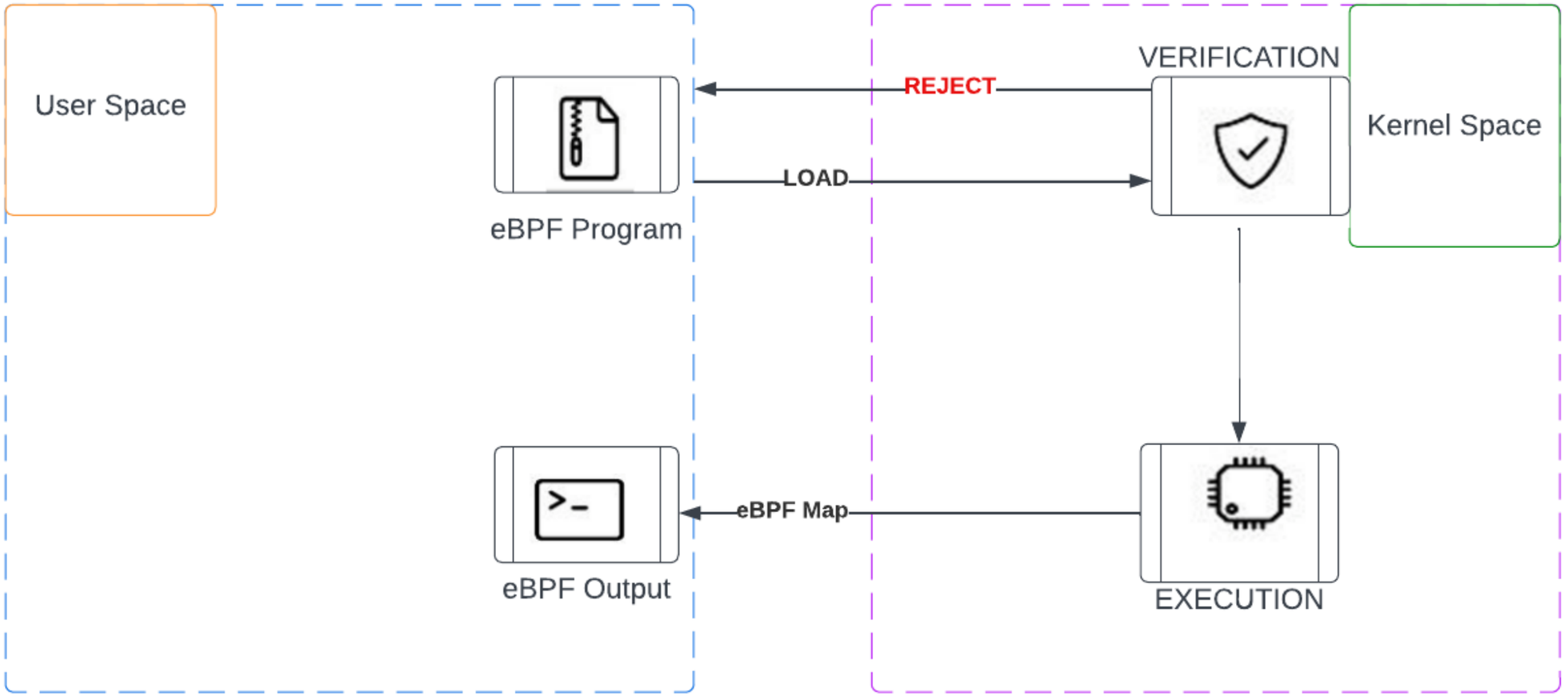


Advantages of eBPF

Amazing Programmability	Excellent visibility and control	Performance
Flexibility	In-depth observability	Low overhead
Security	Fine-grained control	Efficient packet processing
Portability	Real-time monitoring	Reduced latency
Dynamic runtime behavior	Comprehensive analysis	Improved scalability

Demystifying eBPF

An Easy-to-Understand Overview



IP Intelligence

Decoding IP Intelligence

- IP Intelligence module in eBPFshield blocks outgoing connections to blacklisted IPs.
- eBPFshield updates its threat feed list from sources like Talos Intelligence.
- Regular updates of the threat feed list can be scheduled with the update_feed_list script.
- Real-world use cases demonstrate the effectiveness of IP intelligence in preventing access to malicious destinations.

Enabling IP-Intelligence in eBPFShield

Update the threat feed list with the command: **`./update_feeds.sh`**
Start the process with IP Intelligence feature and blocking action: **`python3 main.py --feature ebpf_ipintelligence --block kill`**
Supported actions with the "`--block`" flag: print (**`default`**), **`suspend`**, **`kill`**, **`dump`**

Actions like suspend, kill, and dump can be used to immediately stop potentially malicious behavior.

IP Intelligence

```
$ python3.7 main.py  --feature ebpf_ipintelligence --block kill
Namespace(block='kill', buffer_size=10, bytes_to_capture=100, cport=9000, delay=100,
feature='ebpf_ipintelligence', n_runs=10, qdisc='fq', rate=8, run_scenario='just_one_
flow', store_pcaps=False, time=10, verbose=False)

The program is running. Press Ctrl-C to abort.

Client:b'curl' (pid:390539) was killed by eBPFShield (ip-blacklist:31.3.230.31)

Client:b'curl' (pid:390540) was killed by eBPFShield (ip-blacklist:185.242.113.224)
```

```
$ curl -v 31.3.230.31
*   Trying 31.3.230.31:80...
* TCP_NODELAY set
Killed
$ curl -v 185.242.113.224
*   Trying 185.242.113.224:80...
* TCP_NODELAY set
Killed
$ curl google.com
<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.com/">here</A>.
</BODY></HTML>
$
```

DNS Monitoring

Decoding DNS Monitoring

- Monitors DNS queries in the system
- Helps detect and block DNS tunneling attempts in real-time
- Provides proactive defense against potential DNS-based attacks
- Prevents damage before it occurs by identifying and blocking malicious DNS queries

Enabling DNS Monitoring in eBPFShield

- Start eBPFShield in DNS monitoring mode with the following command:
  `python3 main.py --feature ebpf_monitor`

- Test DNS queries from a client by using the dig command with the following syntax:
  `dig @1.1.1.1 geekwire.com +tcp`

- eBPFShield will capture and display all DNS queries made from the client, providing visibility into DNS traffic for effective monitoring and detection of potential DNS tunneling attempts.

DNS Monitoring

DNS Request and Response

```
root@host-virtualfair:~# dig @1.1.1.1 google.com +tcp +short
142.250.183.174
root@host-virtualfair:~# dig @1.1.1.1 geekwire.com +tcp

; <<>> DiG 9.16.1-Ubuntu <<>> @1.1.1.1 geekwire.com +tcp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59677
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;geekwire.com.                  IN      A

;; ANSWER SECTION:
geekwire.com.           251     IN      A       104.26.14.176
geekwire.com.           251     IN      A       172.67.69.185
geekwire.com.           251     IN      A       104.26.15.176

;; Query time: 32 msec
;; SERVER: 1.1.1.1#53(1.1.1.1)
;; WHEN: Sun Jan 15 17:53:24 UTC 2023
;; MSG SIZE  rcvd: 89

root@host-virtualfair:~#
```

eBPFShield Monitoring DNS Queries

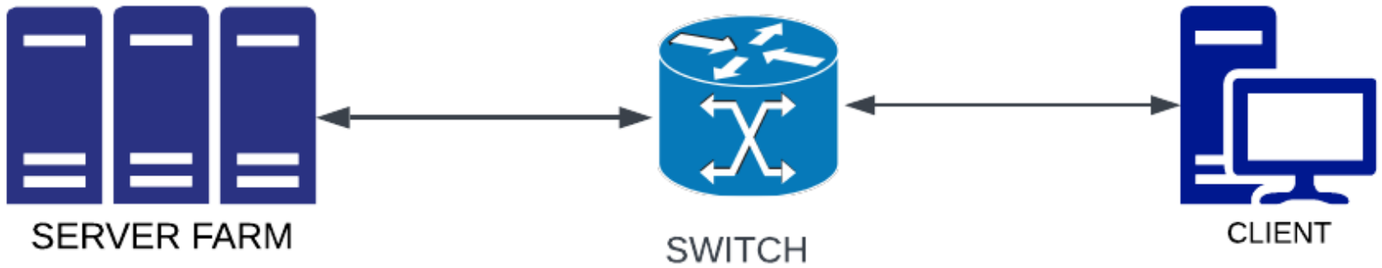
```
root@host-virtualfair:~/arsenal/eBPFShield# python3 main.py --feature ebpf_monitor
The program is running. Press Ctrl-C to abort.
COMM=dig PID=140898 TGID=140899 DEV=ens3 PROTO=TCP SRC=10.218.20.37 DST=1.1.1.1 SPT=44695 DPT=53 UID=0 GID=0 DNS_QR=0 DNS_NAME=google.com. D
NS_TYPE=A
COMM=dig PID=140898 TGID=140899 DEV=ens3 PROTO=TCP SRC=1.1.1.1 DST=10.218.20.37 SPT=53 DPT=44695 UID=0 GID=0 DNS_QR=1 DNS_NAME=google.com. D
NS_TYPE=A DNS_DATA=142.250.183.174
COMM=dig PID=140902 TGID=140903 DEV=ens3 PROTO=TCP SRC=10.218.20.37 DST=1.1.1.1 SPT=41777 DPT=53 UID=0 GID=0 DNS_QR=0 DNS_NAME=geekwire.com.
 DNS_TYPE=A
COMM=dig PID=140902 TGID=140903 DEV=ens3 PROTO=TCP SRC=1.1.1.1 DST=10.218.20.37 SPT=53 DPT=41777 UID=0 GID=0 DNS_QR=1 DNS_NAME=geekwire.com.
 DNS_TYPE=A DNS_DATA=104.26.14.176
COMM=dig PID=140902 TGID=140903 DEV=ens3 PROTO=TCP SRC=1.1.1.1 DST=10.218.20.37 SPT=53 DPT=41777 UID=0 GID=0 DNS_QR=1 DNS_NAME=geekwire.com.
 DNS_TYPE=A DNS_DATA=172.67.69.185
COMM=dig PID=140902 TGID=140903 DEV=ens3 PROTO=TCP SRC=1.1.1.1 DST=10.218.20.37 SPT=53 DPT=41777 UID=0 GID=0 DNS_QR=1 DNS_NAME=geekwire.com.
 DNS_TYPE=A DNS_DATA=104.26.15.176
```

eBPFShield Machine Learning

Decoding and Usage eBPFShield ML

- **Code Compilation**: Wrapper code compiled for userspace and kernelspace with alternative data structures.
- **Decision Engine**: Python module evaluates packets to detect malicious activity in real-time.
- **Experimental Test**: Emulated network with Linux namespaces, showing 10% improvement in analyzed packets.
- **Future Directions**: Explore performance of other machine learning models in eBPF implementation.

```
Run in userspace
- g++ -DUSERSPACE -fpermissive -I/usr/include/bcc ebpf_wrapper.cc -lbcc -o ebpf_wrapper
Run in kernel space
- g++ -fpermissive -I/usr/include/bcc ebpf_wrapper.cc -lbcc -o ebpf_wrapper
```



eBPFShield Forensics

Decoding eBPFShield Forensics

- System Call and Kernel Event Analysis: Detects code injection into other processes.
- Identification of Malicious Files and Processes: Identifies malicious files and processes.
- Demo Use-case #1: Demonstrates code injection techniques used by attackers.
- **Additional Support**: Anti-debugging technique detection, illegitimate shell detection, kernel module loading detection, fileless execution

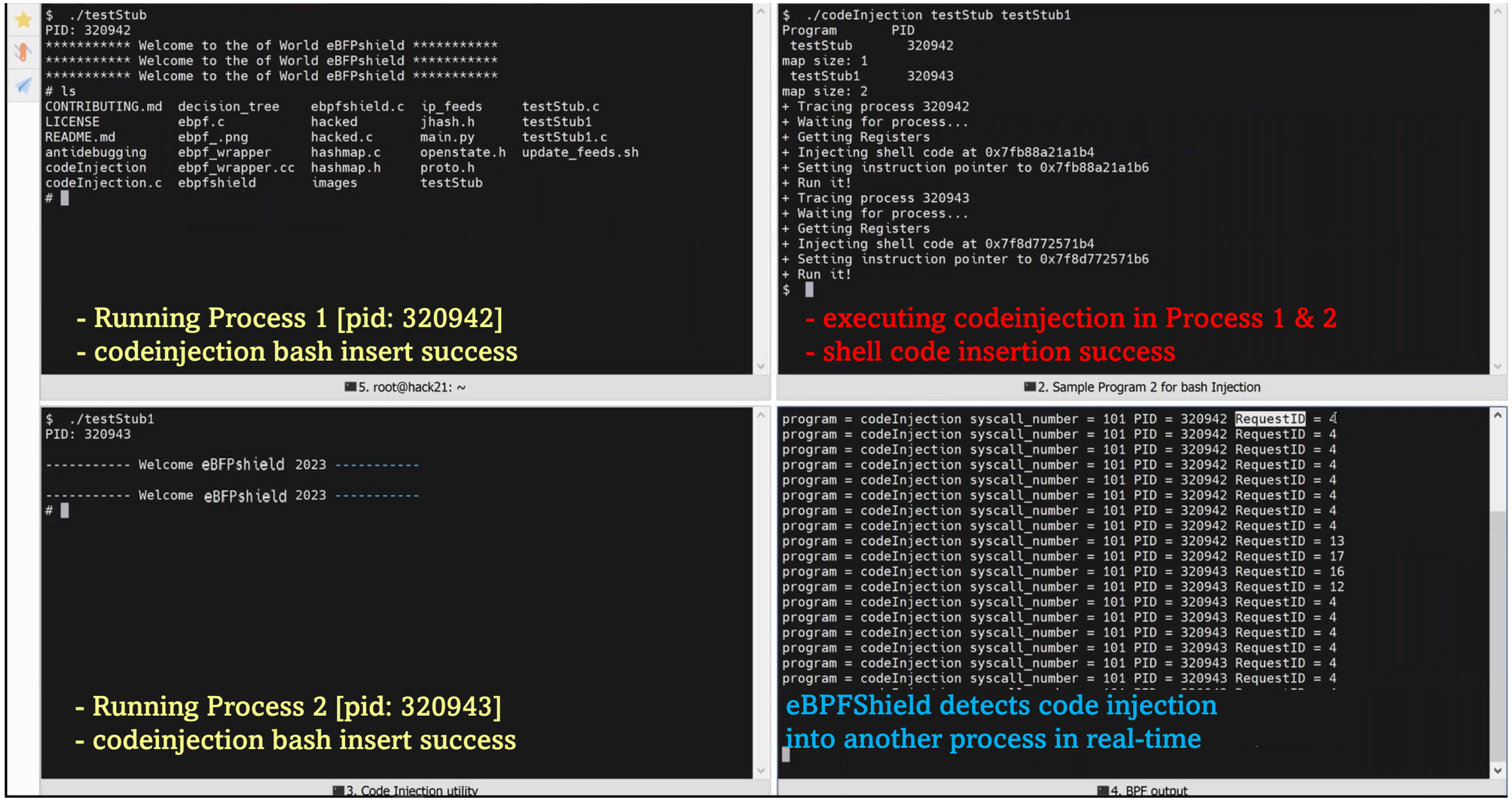
Enabling Forensics in eBPFShield

- Start eBPFShield in Forensics mode with the following command:
  `python3 main.py --feature ebpf_antidebugging`

- Run CodeInjection using
  ./CodeInjection <process 1> <process 2>

- eBPFShield Defense Demo: Real-time detection and prevention of exploits using cutting-edge eBPF technology.

eBPFShield Forensics – In Action



The Road Ahead

Mapping Out Our Future Work

- **Q4 2022**: Released IP-Intelligence & DNS Monitoring; focus on bug fixes and performance improvements; adding new threat intel feeds.
- **Q1 2023**: Adding eBPF Forensics; and eBPF Machine Learning..
- **Q2 2023**: Integrating eBPFShield with popular SIEM systems; adding DNS packet detection on non-standard ports.
- **Q3 2023**: Adding JSON output feature; improving usability with detailed documentation and tutorials.
- **Q4 2023**: Adding action on malicious connections (suspend/kill/process dump); enhancing security with encryption and authentication.
- **Q1 2024**: Expanding support for Windows, macOS, and non-standard DNS ports.

Contributing

When contributing to this repository, please first discuss the change you wish to make via issue, email, or any other method with the owners of this repository before making a change.

Please note we have a code of conduct, please follow it in all your interactions with the project.

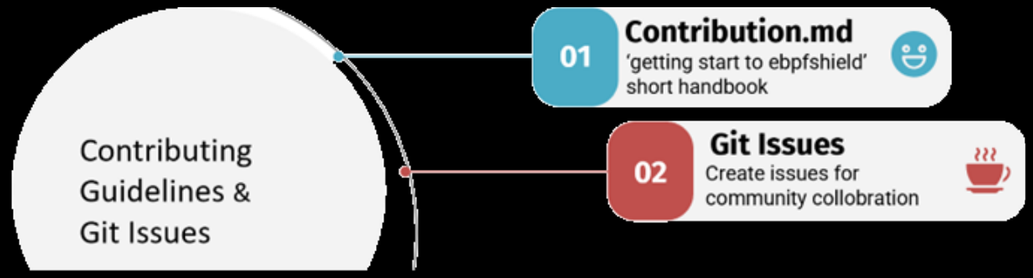
Pull Request Process

1. Ensure any install or build dependencies are removed before the end of the layer when doing a build.
2. Update the README.md with details of changes to the interface, this includes new environment variables, exposed ports, useful file locations and container parameters.
3. Increase the version numbers in any examples files and the README.md to the new version that this Pull Request would represent. The versioning scheme we use is SemVer.
4. You may merge the Pull Request in once you have the sign-off of two other developers, or if you do not have permission to do that, you may request the second reviewer to merge it for you.

Code of Conduct

Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and



github.com/sagarbhure/eBPFShield